

Data Distribution & Processing CSCI  
Data Fusion CSC  
Thor Design Panel 2/3

November 11 , 1997  
Version 2.0

1. Data Fusion.....	3
1.1 Data Fusion Introduction.....	3
1.1.1 Data Fusion Overview .....	3
1.1.2 Data Fusion Operational Description .....	4
1.2 Data Fusion Specifications .....	4
1.2.1 Data Fusion Ground Rules.....	4
1.2.2 Data Fusion Functional Requirements.....	4
1.2.3 Data Fusion Performance Requirements.....	5
1.2.4 Data Fusion Interfaces .....	5
1.2.5 Data Fusion Flow Diagram.....	6
1.3 Data Fusion Design Specification .....	6
1.3.1 Data Fusion Off-line Processing.....	6
1.3.1.1 Data Fusion Algorithm Template Editor .....	6
1.3.1.2 Data Fusion Code Generation .....	7
1.3.1.3 Data Fusion User Instance Files .....	7
1.3.1.4 Data Fusion Input/Output Validation .....	7
1.3.1.5 Data Fusion Executable Build Scripts.....	7
1.3.2 Data Fusion Runtime Processing.....	7
1.3.2.1 Data Fusion Executables .....	7
1.3.2.2 Data Fusion Startup Script .....	7
1.3.3 Data Fusion Detailed Data Flow .....	8
1.3.4 Data Fusion External Interfaces .....	9
1.3.4.1 Data Fusion Message Formats.....	9
1.3.4.2 Data Fusion Display Formats .....	9
1.3.4.3 Data Fusion Instance File Format.....	10
1.3.4.4 Data Fusion Recorded Data.....	10
1.3.4.5 Data Fusion Printer Formats .....	10
1.3.4.6 Data Fusion Inter-process Communications .....	10
1.3.4.7 Data Fusion External Interface Calls.....	10
1.3.5 Data Fusion Internal Interfaces .....	11
1.3.6 Data Fusion Test Plan.....	11
1.3.6.1 Environment.....	11
1.3.6.2 Test Case 1 - Define and build algorithm. ....	11
1.3.6.3 Test Case 2 - View Data Fusion algorithm on the DDP. ....	11
1.3.6.4 Test Case 3 - Value distribution and automatic updates.....	12
1.3.6.5 Test Case 4 - Fusion Processing inhibit and enable. ....	12
1.3.6.6 Test Case 5 - Setting Fusion FD values (calibration constants). ....	12
1.3.6.7 Test Case 6 - Nested Fusion algorithms.....	12

# 1. Data Fusion

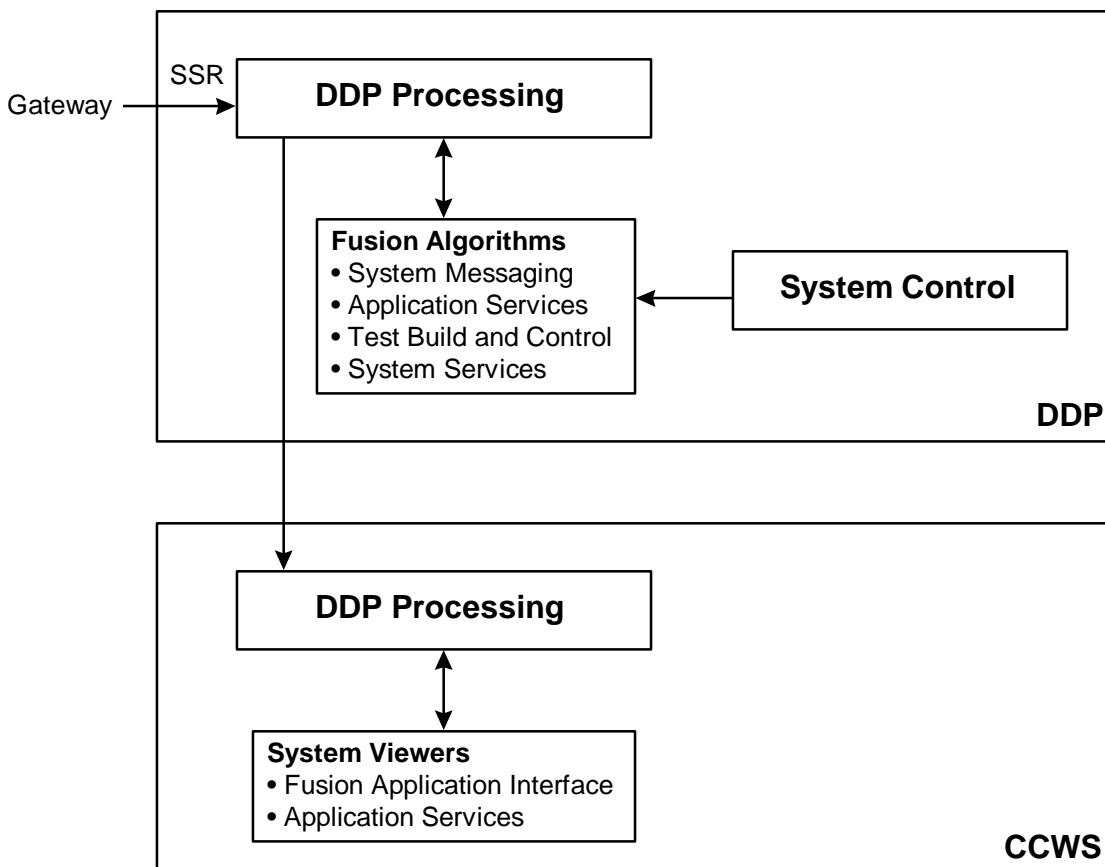
## 1.1 Data Fusion Introduction

### 1.1.1 Data Fusion Overview

The Data Fusion CSC resides in the DDP. Data Fusion provides the capability of running system fusion algorithms to read FD data from Data Distribution via Data Distribution and perform algorithms based on FD measurement values and constants. All inputs will be based on queued change values.

The majority of the Data Fusion work involves off-line scripts to generate algorithms templates, validate algorithm inputs/outputs, and build runtime executables.

The Data Fusion focuses on providing a comprehensive solution to performing data fusion. Requirements and statement of work items pertain to system fusion, but, in some instances may also be applied to pseudo FDs. It is the intent of the Data Fusion Completion Thread to examine pseudo calculations as well as system fusion. Many of the scripts created for system data fusion may also be reused for pseudo calculations.



Data fusion algorithms will run on the DDP and will be started by Autopilot via System Control. Algorithms will run at either a “high” or “normal” priority.

System Viewers will display fusion information and will run on the CCWS. The viewers will retrieve the

algorithm description, inputs, and output from the DDP application interface functions. The viewers will also interface with Data Distribution to retrieve the current fused data value.

### **1.1.2 Data Fusion Operational Description**

Data Fusion supports executables based on C++ algorithms contained in an instance file built by a fusion user defining the inputs, outputs, and priority for a given algorithm.

## **1.2 Data Fusion Specifications**

### **1.2.1 Data Fusion Ground Rules**

1. Data Fusion inputs can be any FD in the TCID and / or a user defined constant.
2. Fusion calculations/formulas may contain user-changeable coefficients. Altering the coefficients will be accomplished through the use of pseudo FDs.
3. Fusion FDs and Fusion algorithms can not be created in a real-time environment. They must be added/changed prior to TCID build.
4. Data Fusion will make use of Data Distribution to obtain and publish fusion data.
5. Data Fusion applications will be executable on the DDP only.
6. Data Fusion algorithms will be coded in C++.
7. Data fusion priority is implemented by the Operating System on a Fusion application basis.

### **1.2.2 Data Fusion Functional Requirements**

The following Data Fusion Functional requirements are listed under two sections:

- Data Fusion off-line processing
- Data Fusion run-time processing

The off-line segment involves:

1. Creating the algorithm class template editor.
2. Creating the viewer text input file based on the editor output.
3. Creating a script which builds the system fusion application in C++.
4. Creating the script which validates the FDs against the valid FDs in the TCID.
5. Ensuring that only change data operations are executed.
6. Creating a script to automate starting the fusion algorithms via System Control or the DDP during runtime.

The run time segment consists of:

1. Executing a fusion algorithm on the DDP.
2. Providing the APIs to allow access to the viewer text input file information.

Requirements carried forward from Redstone

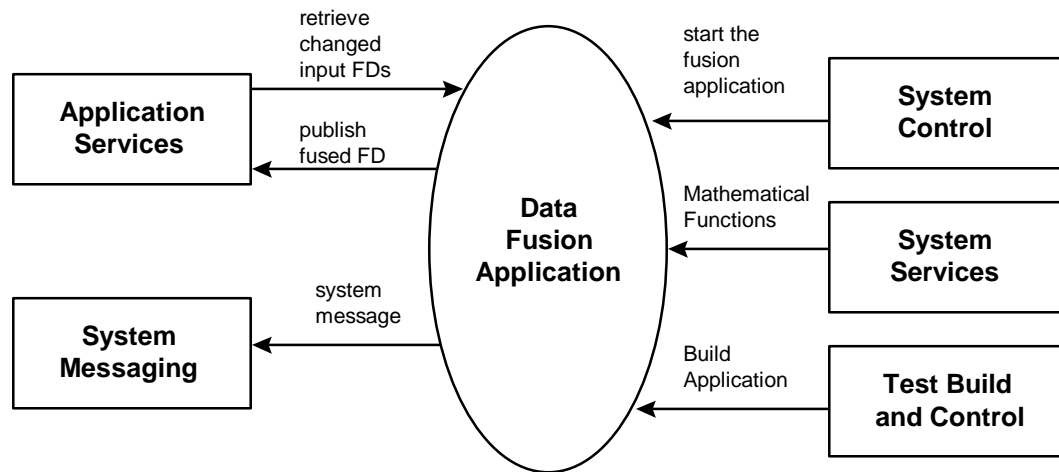
1. Provide capability for performing data fusion with queued FDs.
2. Provide mechanism to prioritize fusion algorithms.

Note: Software issues 215 and 216 for Redstone refer to these two capabilities. Since these issues are now tracked through this Thor DP2/3 document, the Redstone issues will be closed.

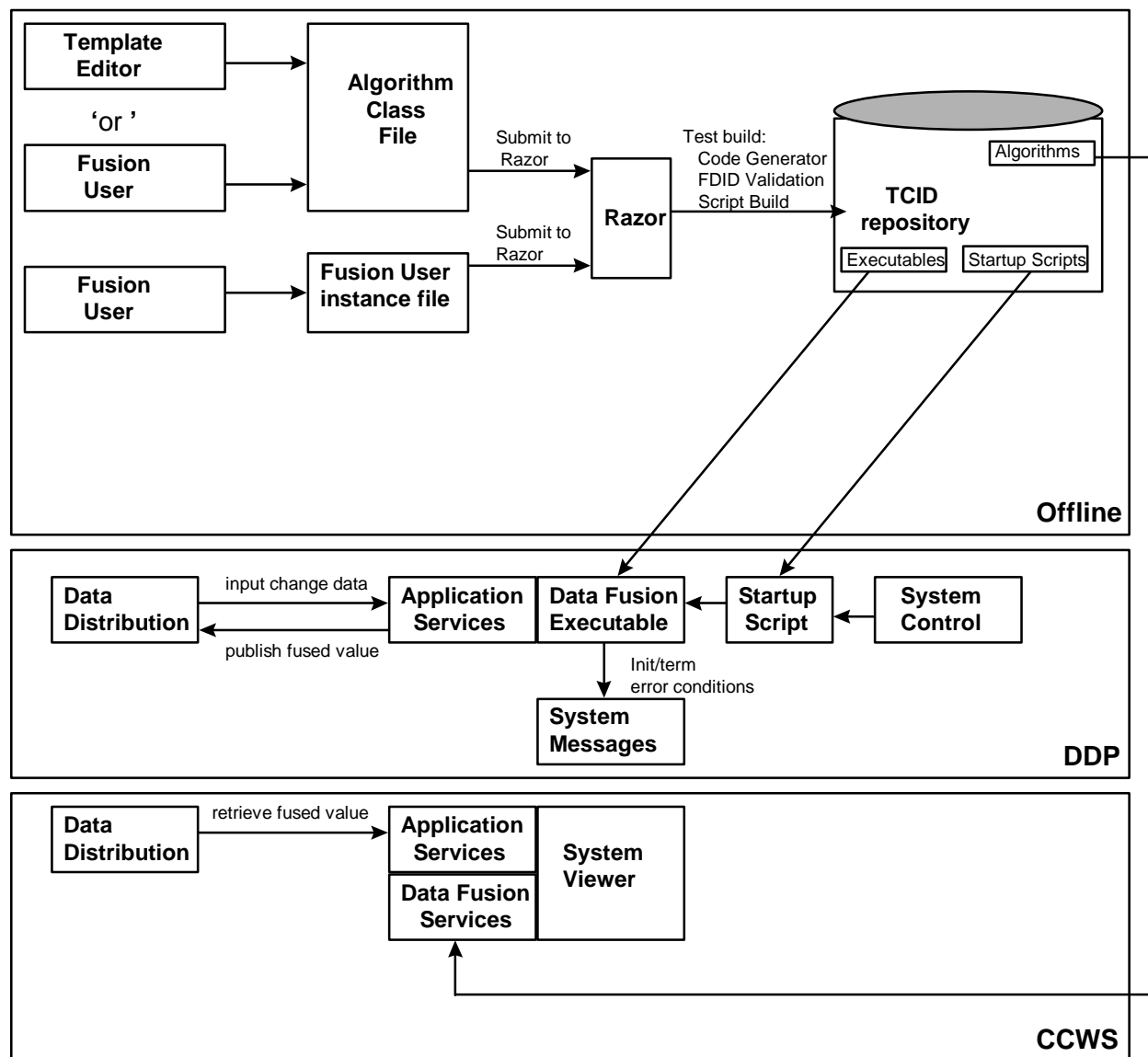
### 1.2.3 Data Fusion Performance Requirements

The goal is to execute the fusion algorithms within one SSR time Period. However, to assure that time critical functions are executed in a timely fashion, Data Fusion will provide a priority mechanism.

### 1.2.4 Data Fusion Interfaces



### 1.2.5 Data Fusion Flow Diagram



## 1.3 Data Fusion Design Specification

Data Fusion supports defining algorithm definitions to be used by the fusion viewer, validating inputs/outputs for fusion FDs, as well as setting up scripts for runtime execution at normal and high priority rates.

### 1.3.1 Data Fusion Off-line Processing

#### 1.3.1.1 Data Fusion Algorithm Template Editor

A template editor will be defined to allow a fusion user to define/generate a generic algorithm to be stored in an algorithm repository. The template will request the necessary information to satisfy system viewer requirements for displaying fusion algorithms. The template editor is optional, but recommended. Once the output of the template is

generated, the user is requested to input the algorithm initialization method and execution method.

#### **1.3.1.2 Data Fusion Code Generation**

A script will be provided which will be run by Test Build and Control to take each of the algorithm files located in the algorithm repository and wrap the requested inputs and outputs with the appropriate Data Distribution function calls. These algorithms will be assembled into a set of Global Fusion applications.

#### **1.3.1.3 Data Fusion User Instance Files**

Fusion users are requested to submit instance files containing the algorithms to be run, along with the input parameters/FDs and fused output/fused FD. This file will be stored in an Instance repository in the TCID and will be used by Test Build and Control to run a script provided by Data Fusion to generate the fusion executable.

#### **1.3.1.4 Data Fusion Input/Output Validation**

Validation of algorithm inputs/outputs is done by a script provided by Data Fusion and run by Test Build and Control. The script will parse the instance repository to validate the requested FDs. The input FD name along with the data type/engineering units will be validated against the TCID. The output fused FD name and data type/engineering units will be validated against the TCID also.

#### **1.3.1.5 Data Fusion Executable Build Scripts**

A script will be provided by Data Fusion to build executable files based on the User instance files defined in the Instance repository. The executables will be placed in the TCID repository.

Two executables will be built for each fusion user Instance File, one for normal priority algorithms, and one for the high priority algorithms. A script will be generated to be used by System Control to start the fusion executables.

### **1.3.2 Data Fusion Runtime Processing**

#### **1.3.2.1 Data Fusion Executables**

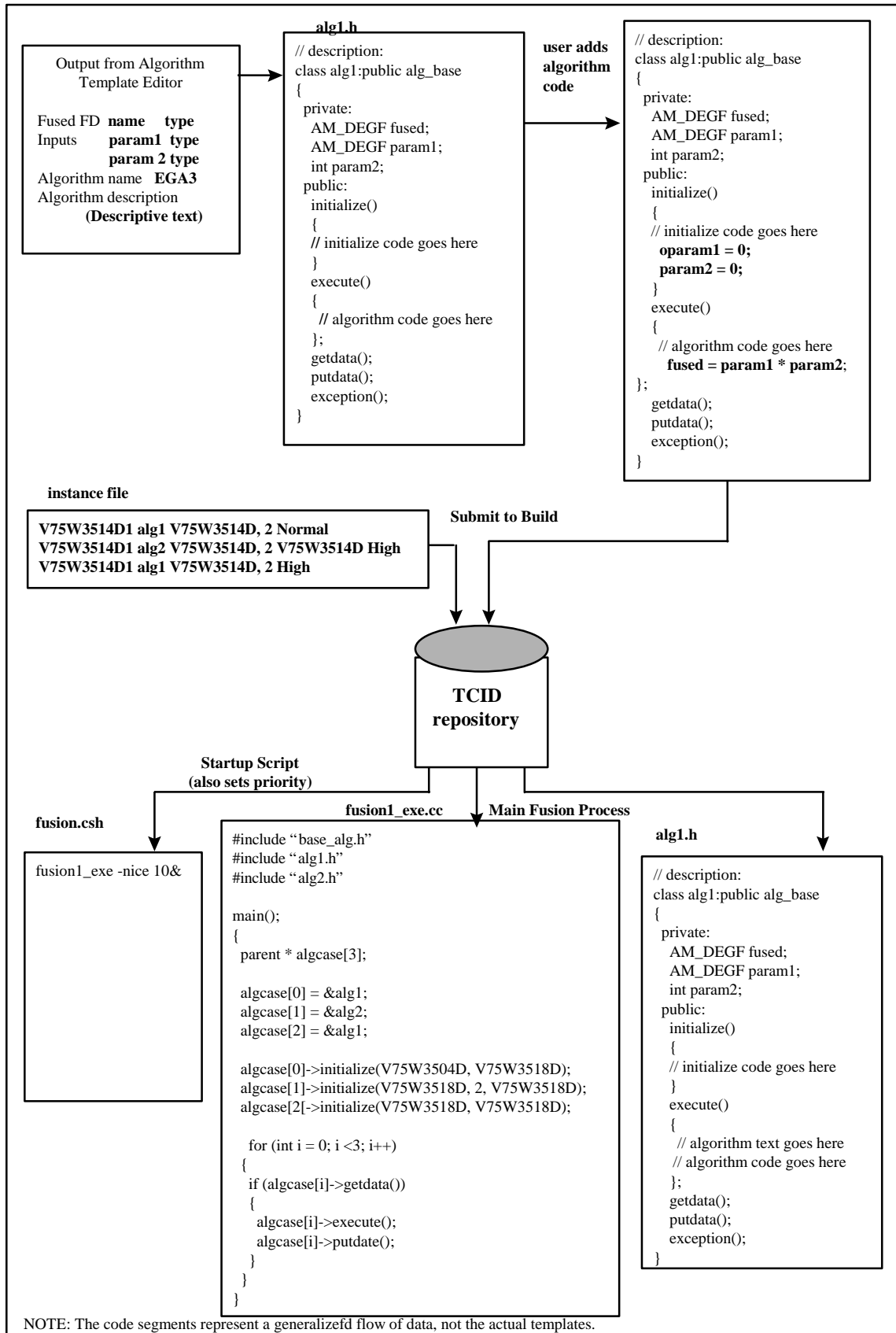
Messages generated by the data fusion executable files will be sent to System Messaging for initialization, error conditions (for example, divide by zero), and termination.

Data fusion executables will run at a normal priority unless the user specifically requested the algorithm run at a high priority. If the algorithm was specified to run at high priority, the data fusion executable will be initialized at a higher priority by invoking the *nice* command during initialization.

#### **1.3.2.2 Data Fusion Startup Script**

Fusion algorithms will be started from System Control by starting the script generated by the Data Fusion. An internal script will be run by Test Build and Control during the build process to define the Startup Script. The Startup Script will contain all the executables that are generated by the build along with the priority that they are to be started at.

### 1.3.3 Data Fusion Detailed Data Flow



**Bold indicates developer written code**

## 1.3.4 Data Fusion External Interfaces

### 1.3.4.1 Data Fusion Message Formats

1. Message Group = DDP  
Severity = Informational

#### **Data Fusion #Argument1# is initialized**

ARGUMENT1 = algorithm name

Help Information Content:  
The algorithm has initialized successfully.

Detailed Information:  
N/A

2. Message Group = DDP  
Severity = Error

#### **Data Fusion #Argument1# had a mathematical exception, errno = #ARGUMENT2#**

ARGUMENT1 = algorithm name  
ARGUMENT2 = system errno

Help Information Content:  
n/a

Detailed Information:  
n/a

3. Message Group = DDP  
Severity = Informational

#### **Data Fusion #Argument1# is terminating**

ARGUMENT1 = algorithm name

Help Information Content:  
n/a

Detailed Information:  
n/a

### 1.3.4.2 Data Fusion Display Formats

Algorithm Template Editor			
File	Help		
Fused FD Type:	<div style="border: 1px solid black; padding: 2px; display: inline-block;">Name</div> <div style="border: 1px solid black; padding: 2px; display: inline-block; margin-left: 10px;">output_fd</div>	<div style="border: 1px solid black; padding: 2px; display: inline-block;">Type</div> <div style="border: 1px solid black; padding: 2px; display: inline-block; margin-left: 10px;">AM_DEGF</div>	
Input Types:	<div style="border: 1px solid black; padding: 2px; display: inline-block;">Name</div> <div style="border: 1px solid black; padding: 2px; display: inline-block; margin-left: 10px;">param1 param2</div>	<div style="border: 1px solid black; padding: 2px; display: inline-block;">Type</div> <div style="border: 1px solid black; padding: 2px; display: inline-block; margin-left: 10px;">AM_DEGF Integer</div>	<div style="border: 1px solid black; width: 20px; height: 40px; margin: 0 auto; position: relative;"> <div style="position: absolute; top: 0; bottom: 0; left: 5px; right: 5px; border: 1px solid black; background: white;"></div> </div>
Algorithm Name:	<div style="border: 1px solid black; padding: 2px; display: inline-block;">EGA3</div>	<div style="border: 1px solid black; padding: 2px; display: inline-block;">Priority</div> <div style="border: 1px solid black; padding: 2px; display: inline-block; margin-left: 10px;">Normal</div>	
Algorithm Description:	<div style="border: 1px solid black; padding: 2px; display: inline-block;">Textual description of the algorithm</div>		<div style="border: 1px solid black; width: 20px; height: 40px; margin: 0 auto; position: relative;"> <div style="position: absolute; top: 0; bottom: 0; left: 5px; right: 5px; border: 1px solid black; background: white;"></div> </div>

Data fusion will provide an optional Algorithm Template Editor to be run in the development environment. This template will generate a file containing the necessary information to be used by the Fusion Viewers. The C++ file generated by the editor will require the user to edit the C++ file and insert the algorithm to be defined.

#### 1.3.4.3 Data Fusion Instance File Format

The user class will submit system fusion applications by creating an ASCII file which includes the following information. The Fused FD name and Input FD names (not constants), will be validated against the TCID during the build process. The priority will indicate to start the process at either normal priority or high priority.

FD Name	Algorithm Class Name	Input FDs and Parameters	Priority
Fused FD	Algorithm	FD1, FD2, P2,P2	High
:	:	: : : :	:
Last Fused FD	Algorithm	FD5,FD6, P5, P6	Normal

#### 1.3.4.4 Data Fusion Recorded Data

There are no input formats for the DDF CSC.

#### 1.3.4.5 Data Fusion Printer Formats

There are no printer formats for the DDF CSC.

#### 1.3.4.6 Data Fusion Inter-process Communications

There is no inter-process for the DDF CSC.

#### 1.3.4.7 Data Fusion External Interface Calls

1. ddf\_id = ddf\_open(fd\_name)  
Open the Algorithm File to retrieve the viewer information.

2. ddf\_get\_desc(ddf\_id)

Retrieve the description for the algorithm.

3. `fdname = ddf_get_output(ddf_id)`  
Retrieve the output/fused FD name for the algorithm.
4. `fdnames[] = ddf_get_inputs(ddf_id,count)`  
Retrieve the list of input FD names/constants for the algorithm.
5. `ddf_close(ddf_id)`  
Close the Algorithm file.

### 1.3.5 Data Fusion Internal Interfaces

There are no internal interfaces for the DDF CSC.

### 1.3.6 Data Fusion Test Plan

#### 1.3.6.1 Environment

Overview: Testing will use two environments:

1. User Fusion Development Environment (Test Case 1)
2. Fusion Execution / Run-Time Environment. (all other Test Cases)

Fusion algorithms will be created to verify the tools and processes provided to create Fusion executable code and viewable Fusion algorithms. These Fusion algorithms will be executed on the DDP using recorded PC-GOAL data. Viewers will be used to verify the values in the CVT for the CCWS.

#### 1.3.6.2 Test Case 1 - Define and build algorithm.

The tools and processes described in the “Data Functional Requirements” diagram in Section 1.2.2 of the Data Fusion CSC Thor Design Panel 2/3 document will be used to create a system data fusion application and submit it to Test Build and Control. A previously-submitted algorithm that was submitted to Test Build and Control and which has been included in the Thor TCID will be executed on the DDP and viewed from a CCWS in Test Case 3.

Dependencies:

- Thor TCID, with Fusion FDs defined
- Development W/S, with Data Fusion tools loaded

#### 1.3.6.3 Test Case 2 - View Data Fusion algorithm on the DDP.

Use the Fusion Viewer to obtain Algorithms on the DDP

Dependencies:

- System Control
- DDP/CCWS platforms
- RTCN and DCN
- Reliable Messaging
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID, with Fusion FDs defined
- Data Distribution
- `cvt_look` for viewing the CVT contents
- Data Fusion Algorithm Viewer

#### **1.3.6.4 Test Case 3 - Value distribution and automatic updates.**

Use the data generator stream which includes the input FDs with changing values. The data needs to provide FD value changes at known time periods, preferably several minutes apart. Execute a Fusion algorithm on the DDP. Verify the values are correct in the CVT on the DDP and CCWS.

Dependencies:

- DDP/CCWS platforms
- RTCN and DCN
- Reliable Messaging
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID, with Fusion FDs defined
- Data Distribution
- cvt\_look for viewing the CVT contents

#### **1.3.6.5 Test Case 4 - Fusion Processing inhibit and enable.**

Use the fusion provided test tool to inhibit/enable fusion output values. Inhibit the publishing of a fused FD in the DDP and verify the CVT does not update the fused value.

Dependencies:

- DDP/CCWS platforms
- RTCN and DCN
- Reliable Messaging
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID, with Fusion FDs defined
- Data Distribution
- cvt\_look for viewing the CVT contents

#### **1.3.6.6 Test Case 5 - Setting Fusion FD values (calibration constants).**

Use the fusion provided test tool to set fusion FD values.

Dependencies:

- DDP/CCWS platforms
- RTCN and DCN
- Reliable Messaging
- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID, with Fusion FDs defined
- Data Distribution
- cvt\_look for viewing the CVT contents

#### **1.3.6.7 Test Case 6 - Nested Fusion algorithms.**

Execute the fusion nested algorithms and verify the output values for one algorithm are the same as the inputs for another algorithm. Verify up to five levels of nesting.

Dependencies:

- DDP platforms
- RTCN and DCN
- Reliable Messaging

- Data Source (PC-GOAL data for STS-TBD)
- Thor TCID, with Fusion FDs defined
- Data Distribution

## APPENDIX A

### Statement of Work

- Provide user guide that lists the logical and mathematical functions provided by Data Fusion.
- Provide the Pre-Build Data Fusion Editor.
- Provide the capability for Fused FDs in the Test Build process.  
*Provided by Test Build and Control.*
- Provide an API for System Viewer with the minimum capability to access Fused FDs including the Fused FD value, associated input FD values, and the function being used to generate the Fused FD.  
*Provided by Data Fusion, Test Build and Control, Application Services, and System Viewers.*
- Provide a FD Design Tool to allow FDs of any type to be added, changed, or deleted. Allow these changes to be made to an Online Data Bank without using DBSAFE. Desktop Debug Environment ONLY)  
*Provided by Test Build and Control.*
- Create Enumerated FD data type supporting up to 256 states.  
*Provided by Test Build and Control and Application Services.*
- Provide performance data for Fusion overhead and functions for system modeling.  
*Data Fusion provides the data to the Performance Evaluation Support group to be analyzed.*
- Provide the capability for the Data Fusion function to be initialized in both Operational and Desktop Debug Environment Configurations.  
*Desktop Debug script will be the same as for the operational system. If Desktop Debug Environment requires changes to the script, the changes will be done manually for Thor.*
- Coordinate with possible baseline changes for allocation of Data Fusion.
- Provide logging of error, performance, and state change information.
- Baseline system messages using the System Message Catalog to include message and help text.

### Requirements from SLS

- 2.2.5.3.1 The following are a list of the SLS requirements pertaining to Data Fusion. The italicized text describes the interpretation or implementation which will satisfy the SLS requirement. CLCS shall provide the capability to define, view, and execute the algorithms for performing data fusion.  
*The importance of this requirement is to provide a method for defining and viewing the algorithms used to calculate fused FDs.*
- 2.2.5.3.2 A fused Data Function Designator shall be recalculated whenever any of its input parameters change.  
*This requirement implies that the fusion will operate at the change data rate for system FDs.*
- 2.2.5.3.3 When the value of a Fused FD changes, the new value shall be re-transmitted to all users, and system or user applications at the System or Display Synchronous Rates.  
*This requirement is satisfied by Data Distribution distributing any FD change value.*
- 2.2.5.3.4 The Data Fusion function shall allow activation and deactivation of fused Data Function Designator Processing.  
*Interpreting “deactivation” to mean “inhibit”, this requirement is satisfied by Data Distribution activation/inhibiting all FDs including pseudo FDs.*
- 2.2.5.3.5 The Data Fusion function shall provide the capability to set the value of a Fused FD.  
*This is interpreted as the ability to publish the value of a fused FD, which is satisfied by Data Distribution. The requirement to provide a debug capability to overwrite an FD is part of Data Distribution.*
- 2.2.5.3.6 The RTPS Measurement FD Fusion function shall be fault tolerant.  
*System fusion fault tolerance will be handled by DDP redundancy and System integrity. DDP redundancy and System Integrity will not be done for Thor. Algorithm fault tolerance will be handled TBD.*
- 2.2.5.3.7 The RTPS Measurement FD Fusion function shall provide up to 5 levels of Data Fusion nesting.  
*A minimum of 5 levels will be tested for adherence to this requirement. However, levels of nesting will not be*

*programmatically limited. It is up to the application program to code the required nesting. A data fusion pre-build script will be run to provide a check to ensure the application does not have circular dependencies for published FDs.*

- 2.2.5.3.8 RTPS shall provide the capability to automatically maintain the values of Data Fusion FDs. *It is assumed that fused FDs will be treated as any other FD. The capability to maintain fused data values is provided by Data Distribution.*
- 2.2.5.3.9 the RTPS Measurement FD Fusion function shall provide a mechanism to prioritize fusion calculations. *Priorities can be handled by the application or utilizing the UNIX priority system for establishing the priority. This capability will not be incorporated in the Thor delivery.*
- 2.2.10.1.7 The RTPS shall record all Fused Data FD and Fused FD parameter changes to the SDC. *This requirement is implemented by Data Distribution.*

### **Performance Requirements from SLS**

- 2.2.2.1.15 The Data Fusion function shall support the “system maximum band width” with one fusion calculation per end item FD change. SLS rationale: Although our goal is to execute the function algorithms within one System Synchronous Rate Time Period, no limit requirement is included. During peak rate times, the algorithm must support a lag in fusion calculations. However, to assure that time critical functions are executed in a timely fashion, Data Fusion will provide a priority mechanism (reference Section 2.2.5 - System Support for User applications/Data Fusion).

### **SOW items carried forward from Redstone**

- Provide capability for performing data fusion with queued FDs.
- Provide mechanism to prioritize fusion algorithms.

Note: Software issues 215 and 216 for Redstone refer to these two capabilities. Since these issues are now tracked through this Thor DP2/3 document, the Redstone issues will be closed.